Generalized K-fan Multimodal Deep Model with Shared Representations

Gang Chen and Sargur N. Srihari *
Computer Science Dept.
University at Buffalo, SUNY
Buffalo, NY 14260

Abstract

Multimodal learning with deep Boltzmann machines (DBMs) is an generative approach to fuse multimodal inputs, and can learn the shared representation via Contrastive Divergence (CD) for classification and information retrieval tasks. However, it is a 2-fan DBM model, and cannot effectively handle multiple prediction tasks. Moreover, this model cannot recover the hidden representations well by sampling from the conditional distribution when more than one modalities are missing. In this paper, we propose a K-fan deep structure model, which can handle the multi-input and muti-output learning problems effectively. In particular, the deep structure has K-branch for different inputs where each branch can be composed of a multi-layer deep model, and a shared representation is learned in an discriminative manner to tackle multimodal tasks. Given the deep structure, we propose two objective functions to handle two multi-input and multi-output tasks: joint visual restoration and labeling, and the multi-view multi-calss object recognition tasks. To estimate the model parameters, we initialize the deep model parameters with CD to maximize the joint distribution, and then we use backpropagation to update the model according to specific objective function. The experimental results demonstrate that the model can effectively leverages multi-source information and predict multiple tasks well over competitive baselines.

1 Introduction

We are exploring the multimodal learning in a joint framework when we have multiple forms of data available in the information age, such as images, labels, texts and videos. Each modality is characterized by very distinct statistical properties, but it also reflects one or two facets of the data even though they come from different input channels. Thus, it is possible to leverage different inputs to learn a shared representation in the prediction tasks, such as data restoration and classification. Recent advances in deep learning [1] and multi-modality learning [2] shed lights on joint representation learning which captures the real-world concept that the data corresponds to. The deep learning methods [3, 4], such as deep belief networks (DBNs) [3], CNN [6, 7] and LSTM [8], can learn an abstract and expressive representations, which can capture a huge number of possible input configurations. Hence, the representation learned is useful for classification and information retrieval. The multimodal learning model [5] in a sense extends the deep learning framework, such as deepautoencoder or deep Boltzmann machines (DBMs), to handle different modalities. Thus it can learn a joint representation such that similarity in the code space implies similarity of the corresponding concepts. However, these previous multi-modality models [2, 5] are kind of 2-fan deep model, and can only handle or predict one task. Often, the joint representation learned is not robust enough when the data is typically very noisy and there may be missing. Furthermore, how to leverage multi-source information from multimodalities is also an interesting topic for classification and information retrieval.

In this paper, we propose a K-fan deep structure model, where we generalize the previous 2-fan multimodal learning [2, 5] to handle multiple inputs and outputs. Our model is composed of K-branch deep models, with a shared hidden layer. In particular, the deep structure model has K pathway for different inputs respectively, and learn a shared representation to tackle multimodal tasks in an discriminative manner. Our model is powerful because each branch can be a multi-layer deep model, for example, we can use DBN, CNN or LSTM in each branch to handle different modalities, such as images, texts and videos. Refer to the right panel in Fig. 1 with DBNs used in each branch for a 3-fan deep structure case. Most similar to our work are

*Footnote for author to give an alternate address.

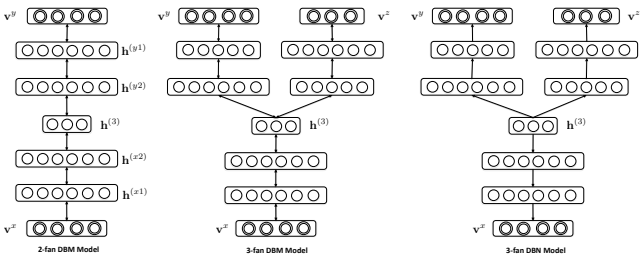


Figure 1: The left is an example of 2-fan multimodal DBM, with shared representations via the 3-layer deep architecture. The middle is a 3-fan DBM, with a joint representation shared by $\mathbf{v}^x$, $\mathbf{v}^y$ and $\mathbf{v}^z$. The right is a 3-fan DBN, which is different from DBM and can be easily generalized into K-fan deep structure by extending more branches from the shared representations. Note that except the shared hidden nodes, each branch can be different deep models, such as DBN, CNN and LSTM, with different number of layers and also different number of nodes in each layer.

the bi-modal deep models [2, 5] to handle image-text data or speech-vision data. The multimodal DBM proposed by Ngiam et al. [2] used a deep autoencoder for speech and vision fusion, while the other method [5] leveraged DBMs to learn hidden representations for bi-modal image-text data. There are, however, several crucial differences between our model and other methods. First, in this work we can handle the K-fan different modalities, instead of bi-modal inputs. Moreover, each branch can be a deep model, such as DBN, CNN and LSTM to handle different multimodalities. Thus, our model is more powerful than 2-fan modality models. Secondly, our deep structure can jointly learn from multiple inputs to predict multi-output with shared representations. Lastly, given the deep K-fan structure, it is very flexible to design an objective function and learn the model parameters in an discriminative manner by fusing multiple inputs.

In this paper, we use a composition of multiple DBNs as a special case in our K-fan model. Note that other deep model such as CNN and LSTM can be used in each branch to handle more complex data. In the learning stage, all inputs are thought as observed data. Thus, we pretrain the model by leveraging all channels to learn the joint representations via Contrastive Divergence (CD). Then, we finetune the model parameters via backpropagation according to different tasks. In the inference stage, our model uses the feed-forward to handle multiple inputs and output predictions.

We test our model on two different tasks: joint visual restoration and labeling, as well as multiclass object recognition task. As for the first experiment, it is a multi-task learning problem, which need to jointly restore the data as well as label it. While for the second experiment, we leverage multiple inputs or resources to improve the prediction accuracy. The experimental results demonstrate the advantages of our model over other competitive baselines, such as multimodal DBM and support vector machines (SVM).

2 Related work

Over the past few years, there have been several approaches proposed to learning from multimodal data. For example, a joint model of images and text using dual-wing harmoniums is builded by Xing et al. [9], which is a generative model and can be viewed as a linear RBM model with Gaussian and Poisson visible units. Huiskes et al. [10] used standard low-level image features with additional captions, or tags, to improve classification accuracy significantly over SVM. A similar approach [11], based on multiple kernel learning framework, was also proposed and demonstrated that an additional text modality can improve the accuracy of SVMs on various object recognition tasks.

Recently, the multimodal learning with deep learning has attracted great attention in machine learning community. The approach of Ngiam et al. [2] used a deep autoencoder for speech and vision fusion. And a multimodal DBM [5] is also proposed, which can be viewed as a composition of unimodal undirected pathways. Each pathway can be pretrained separately in a completely unsupervised fashion, with a large supply of unlabeled data. Any number of pathways each with any number of layers could potentially be used. One advantage of the multimodal DBM is that it is a generative model, which allows the model to naturally handle missing data in one channel by sampling. However, it cannot effectively handle data missing in multiple channels. More specifically, it cannot be used to predict multiple tasks.

In this paper, we propose a unified K-fan deep neural network, which can learn a shared representation to handle different tasks. Moreover, each branch can be composed of a multi-layer deep model, such as CNN, DBN and LSTM to handle different inputs. Thus, our model is more powerful by generalizing previous multimodal models [2, 5] for more complex tasks. We initialize the model parameters in a generative manner with CD algorithm. While in finetuning stage, we can update the model parameters by optimizing the given objective functions. Thus, our model can leverage multiple resources and also handle or predict multiple outputs. We test our model on two different tasks: joint visual restoration and recognition, and multi-view multi-class object recognition.

As for the former task, usually, the visual restoration and recognition were addressed in separated pipeline, for example image denoising followed with recognition. One related work is denoising autoencoder [12], which extends the work [13, 1] and minimizes the reconstruction loss between the input (the corrupted data) and the output (the clean version of it), to learn feature representation. Recently, a robust Boltzmann machine (RoBM) [14] was in-

troduced for recognition and denoising. This model added another shape RBM to the Gaussian RBM prior to model the noise variables which indicate where to ignore the occluder in the image. Thus, the RoBM contains a multiplicative gating mechanism to handle unexpected corruptions of the observed variables. However, the experiments only show its effectiveness for regular structural noise.

As for the multi-view multi-class object recognition, Torralba et al. [15] proposed the joint boosting method to learn shared representations for multi-view multi-class object detection. Huiskes et al. [10] used SVMs to leverage the additional view information to boost the classification performance.

We compared our method to other competitive baselines such as multimodal DBM and SVMs, and the experimental results show the advantages of our model over a variety of tasks.

3 K-fan multimodal deep model

Our K-fan deep structure is composed of K deep models in each branch, with a shared representation to handle multimodalities. For clarity, we use the DBNs as the deep model in each branch in our K-fan deep structure to explain our model. More specifically, we use restricted Boltzmann machines (RBMs) as the building blocks in each branch. In the pretraining stage, we initialize the parameters with CD algorithm as the deep Boltzmann machines. In the finetuning stage, our model is more like the generalized deep autoendcoder with multiple pathways. In the following parts, we will review RBMs first, and then we will introduce our model.

3.1 Background

An RBM with n hidden units is a parametric model of the joint distribution between a layer of hidden variables $\mathbf{h} \in \{0,1\}^n$ and the observation $\mathbf{v} \in \{0,1\}^D$. The RBM joint likelihood takes the form:

$$p(\mathbf{v},\mathbf{h}) \propto e^{-E(\mathbf{v},\mathbf{h})} \tag{1}$$

where the energy function is

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{h}^T\mathbf{W}\mathbf{v} - \mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} \tag{2}$$

And we can compute the following conditional likelihood:

$$p(\mathbf{v}|\mathbf{h}) = \prod_i p(v_i|\mathbf{h}) \tag{3a}$$

$$p(v_i = 1|\mathbf{h}) = \text{logistic}(b_i + \sum_j W(i,j)h_j) \tag{3b}$$

$$p(h_i = 1|\mathbf{v}) = \text{logistic}(c_i + \sum_j W(j,i)v_j) \tag{3c}$$

where $\text{logistic}(x) = 1/(1+e^{-x})$. To learn RBM parameters, we need to minimize the negative log likelihood $-\log p(\mathbf{v})$ on training data, the parameters updating can be calculated with an efficient stochastic descent method, namely contrastive divergence (CD) [3]. Thus, we get the following stochastic gradient for $\mathbf{W}$ from CD,

$$\frac{\partial \log p(\mathbf{v})}{\partial W_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{4}$$

where we ignore the biases of both hidden and observation layers. And update θ until convergence with gradient descent

$$\theta = \theta + \eta \frac{\partial \log p(\mathbf{v})}{\partial \theta} \tag{5}$$

where θ is the weight and biases, and η is the learning rate.

A deep (restricted) Boltzmann machine (DBM) is a stack of RBMs, in which each layer can capture complicated and higher-order correlations between the activities of hidden features in the layer below [16]. For a two layer DBM, it contains one layer visible units $\mathbf{v} \in \{0,1\}^D$ and two layer hiddens variables $\mathbf{h}^{(1)} \in \{0,1\}^{n_1}$ and $\mathbf{h}^{(2)} \in \{0,1\}^{n_2}$. The energy of the joint configuration $\{\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}\}$ is defined as (ignoring bias terms):

$$E(\mathbf{v},\mathbf{h};\theta) = -\mathbf{v}^T\mathbf{W}^{(1)}\mathbf{h}^{(1)} - {\mathbf{h}^{(1)}}^T\mathbf{W}^{(2)}\mathbf{h}^{(2)} \tag{6}$$

where $\mathbf{h} = \{\mathbf{h}^{(1)}, \mathbf{h}^{(2)}\}$ represent the set of hidden units, and $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}\}$ are model parameters, representing visible-to-hidden and hidden-to-hidden symmetric interaction terms. Similar to RBMs, this binary-binary DBM can be easily extended to modeling dense real-valued or sparse count data, which has been extensively discussed in [5]. Then, we can get the following likelihood by marginalizing out $\mathbf{h}$

$$\begin{aligned} P(\mathbf{v};\theta) &= \sum_{\mathbf{h}^{(1)},\mathbf{h}^{(2)}} P(\mathbf{v},\mathbf{h}^{(1)},\mathbf{h}^{(2)};\theta) \\ &= \frac{1}{\mathcal{Z}(\theta)} \sum_{\mathbf{h}^{(1)},\mathbf{h}^{(2)}} \exp\big(-E(\mathbf{v},\mathbf{h};\theta)\big) \end{aligned} \tag{7}$$

where $\mathcal{Z}(\theta)$ is the partition function. The parameters in Eq. 7 can be initialized with a greedy layer-wise pretraining step [3]. Basically, it is to learn the current parameters of RBMs with CD, and the learned features of the current layer RBM are treated as the "data" to train the next RBM in the stack. The joint representation learning of multiple inputs, as well as model parameter updating via mean-field will be introduced in the next part.

3.2 The description of joint representations

Our K-fan deep model is a deep neural network with K different kinds of inputs coupled stochastic binary hidden units in a hierarchical structure. The inputs can be binary or real values, and they share a hidden layer via multi-layers

non-linear transform of RBMs for each input. For clarity, we will use 3-way deep structure to explain our model, shown in Fig. 1.

Suppose we have a set of visible inputs $\mathbf{v}^x \in \{0,1\}^D$, $\mathbf{v}^y \in \{0,1\}^D$ and $\mathbf{v}^z \in \{0,1\}^K$, and a sequence of layers of hidden units for each input. Note that $\mathbf{v}^x$, $\mathbf{v}^y$ and $\mathbf{v}^z$ can have different dimensionalities. For clarity, we start by modeling each input using a separate two-layer DBM. For input $\mathbf{v}^x$, and its two layers of hidden units $\mathbf{h}^{(x1)} \in \{0,1\}^{n_{x1}}$ and $\mathbf{h}^{(x2)} \in \{0,1\}^{n_{x2}}$, the probability for the visible vector $\mathbf{v}^x$ is given by

$$P(\mathbf{v}^x;\theta_x) = \sum_{\mathbf{h}^{(x1)},\mathbf{h}^{(x2)}} P(\mathbf{v}^x,\mathbf{h}^{(x1)},\mathbf{h}^{(x2)};\theta_x) \tag{8}$$

$$= \frac{1}{\mathcal{Z}(\theta_x)} \sum_{\mathbf{h}^{(x1)},\mathbf{h}^{(x2)}} \exp\Big(\sum_{k,j} W^{(x1)}_{k,j} v^x_k h^{(x1)}_j + \sum_{j,l} W^{(x2)}_{j,l} h^{(x1)}_j h^{(x2)}_l\Big) \tag{9}$$

where $\theta_x = \{\mathbf{W}^{(x1)}, \mathbf{W}^{(x2)}\}$. Note that we only consider the binary observation (can be easily extended into real value case with Gaussian RBMs) and ignore biases for both visible and hidden units for clarity. Analogously, we can get the likelihoods for $\mathbf{v}^y$ and $\mathbf{v}^z$ respectively in the same formulas, but with different subscripts.

To form our 3-fan deep model, we combine the three models from $\mathbf{v}^x$, $\mathbf{v}^y$ and $\mathbf{v}^z$, by adding an additional layer of binary hidden units on top of them. The resulting graphical model is shown in the right panel of Fig. 1. The joint distribution over the multi-modal inputs can be written as:

$$P(\mathbf{v}^x,\mathbf{v}^y,\mathbf{v}^z;\theta) = \sum_{\mathbf{h}^{(x2)},\mathbf{h}^{(y2)},\mathbf{h}^{(z2)},\mathbf{h}^{(3)}} P(\mathbf{h}^{(x2)},\mathbf{h}^{(y2)},\mathbf{h}^{(z2)},\mathbf{h}^{(3)})$$
$$\Big(\sum_{\mathbf{h}^{(x1)}} P(\mathbf{v}^x,\mathbf{h}^{(x1)},\mathbf{h}^{(x2)})\Big)\Big(\sum_{\mathbf{h}^{(y1)}} P(\mathbf{v}^y,\mathbf{h}^{(y1)},\mathbf{h}^{(y2)})\Big)$$
$$\Big(\sum_{\mathbf{h}^{(z1)}} P(\mathbf{v}^z,\mathbf{h}^{(z1)},\mathbf{h}^{(z2)})\Big) \tag{10}$$

where $\mathbf{W}^{(x3)}$, $\mathbf{W}^{(y3)}$ and $\mathbf{W}^{(z3)}$ are respectively the top layer weights which connected to the top shared layer $\mathbf{h}^{(3)}$ for each input. And the parameters $\theta = \{\theta_x, \theta_y, \theta_z, \mathbf{W}^{(x3)}, \mathbf{W}^{(y3)}, \mathbf{W}^{(z3)}\}$, where we ignore the biases.

3.3 Learning and Inference

In learning stage, all inputs are available. Thus, we can use CD to learn the shared presentation and model parameter effectively. In practice, we divide the parameter estimation into two stages: parameter initialization and fine-tuning. And the parameter initialization stage focuses on learning the joint representations shared by different modalities, while the fine-tuning stage emphasizes on the discriminative learning according to the properties of tasks in our hand. More specifically, because of the joint multimodal deep structure, we first initialize the model parameters by maximizing the joint likelihood in Eq. 10. Then we update our model parameters by optimizing different objective functions according to different tasks. Note that for different functions, we have the same parameter initialization step via CD because we use the same multimodal deep structure.

3.3.1 Parameter initialization

We first use pretraining to initialize the weights of each branch separately. Basically, it is to learn a stack of RBMs greedily layer-by-layer. To put it simply, the learned features of the current layer RBM are treated as the "data" to train the next RBM in the stack. Assume that we have a training set $\mathcal{D} = \langle \mathbf{v}^x_i, \mathbf{v}^y_i, \mathbf{v}^z_i \rangle_{i=1}^N$. Then, for each branch $\mathbf{v}^x$, we can use RBMs to initialize the weights $\{\mathbf{W}^{(x1)}, \mathbf{W}^{(x2)}, \mathbf{W}^{(x3)}\}$ in the layer-wise manner mentioned above.

Then, we update the model parameters with CD. Because this model is intractable, we use an efficient approximate learning and inference [16], such as mean-field method, to estimate data-dependent expectations, and an MCMC based stochastic approximation procedure to approximate the model's expected sufficient statistics. In variational learning [17, 18, 16] the true posterior distribution over latent variables $p(\mathbf{h}|\mathbf{v};\theta)$ for each training vector $\mathbf{v}$, is replaced by an approximate posterior $q(\mathbf{h}|\mathbf{v};\mu)$ and the parameters are updated by following the gradient of a lower bound on the log-likelihood:

$$\ln P(\mathbf{v};\theta) \geq \sum_{\mathbf{h}} q(\mathbf{h}|\mathbf{v};\mu) \ln p(\mathbf{v},\mathbf{h};\theta) + \mathcal{H}(q) \tag{11}$$

$$= \ln p(\mathbf{v};\theta) - KL[q(\mathbf{h}|\mathbf{v};\mu)||q(\mathbf{h}|\mathbf{v};\mu)] \tag{12}$$

where $\mathbf{h} = \{\mathbf{h}^{(x1)}, \mathbf{h}^{(x2)}, \mathbf{h}^{(y1)}, \mathbf{h}^{(y2)}, \mathbf{h}^{(z1)}, \mathbf{h}^{(z2)}, \mathbf{h}^{(3)}\}$, $\mathbf{v} = \{\mathbf{v}^x, \mathbf{v}^y, \mathbf{v}^z\}$, and μ is the mean-field approximation to hidden variable (see further), and $\mathcal{H}(\cdot)$ is the entropy functional. To maximize the log-likelihood of the training data, is to find parameters that minimize the Kullback–Leibler divergences between the approximating and true posteriors.

Similar to [5], we use the naive mean-field approach, with fully factorized distribution to approximate the true posterior:

$$q(\mathbf{h}|\mathbf{v};\mu) = \Big(\prod_i q(h^{(3)}_i|\mathbf{v})\Big)\Big(\prod_k q(h^{(x1)}_k|\mathbf{v}) \prod_j q(h^{(x2)}_j|\mathbf{v})\Big)$$
$$\Big(\prod_k q(h^{(y1)}_j|\mathbf{v}) \prod_j q(h^{(y2)}_j|\mathbf{v})\Big)\Big(\prod_k q(h^{(z1)}_k|\mathbf{v}) \prod_j q(h^{(z2)}_j|\mathbf{v})\Big) \tag{13}$$

where $\mu = \{\mu^{(1)}_x, \mu^{(2)}_x, \mu^{(1)}_y, \mu^{(2)}_y, \mu^{(1)}_z, \mu^{(2)}_z, \mu^{(3)}\}$ are the

mean-field parameters with

$$q(h^{(xl)} = 1) = u_x^{(l)}, \text{for } l = 1, 2. \quad (14)$$

$$q(h^{(yl)} = 1) = u_y^{(l)}, \text{for } l = 1, 2. \quad (15)$$

$$q(h^{(zl)} = 1) = u_z^{(l)}, \text{for } l = 1, 2. \quad (16)$$

$$q(h^{(3)} = 1) = u^{(3)}. \quad (17)$$

Then μ can be used to update the hidden variables in the data dependent item in Eq. 4. And the model dependent hidden variables in Eq. 4 can be sampled with MCMC. Then, the model parameter can be updated with CD algorithm according to Eq. 5.

3.3.2 Parameter fine-tuning

The parameter fine-tuning stage is different from previous methods, and is determined by the tasks that we pursue in our hand. In our experiments, we test our model on two kind of tasks: visual restoration and labeling, and multi-view multi-class object recognition. For the two tasks, we used the same deep structures with K=3, but with different objective functions.

Joint visual restoration and labeling: For the given corrupted input $\mathbf{v}^x$, we need to restore its clear image $\mathbf{v}^y$ as well as predict its label $\mathbf{v}^z$. Thus, we propose the following objective function for the binary case

$$\begin{aligned}\theta &= \text{argmin}_\theta \mathcal{J}(\mathbf{v}^x, \mathbf{v}^y, \mathbf{v}^z; \theta) \\ &= \text{argmin}_\theta - \sum_{i=1}^{N} \mathbf{v}_i^y \log \hat{\mathbf{v}}_i^y + (1 - \mathbf{v}_i^y)\log(1 - \hat{\mathbf{v}}_i^y) \\ &- \lambda \sum_{i=1}^{N} \mathbf{v}_i^z \log \hat{\mathbf{v}}_i^z + (1 - \mathbf{v}_i^z)\log(1 - \hat{\mathbf{v}}_i^z) \quad (18)\end{aligned}$$

where θ is the set of weights in the 3-way deep architecture respectively (we ignore the subscripts for clarity), and λ is the constant to balance the two losses in Eq. 18. And $\hat{\mathbf{v}}_i^y$ and $\hat{\mathbf{v}}_i^z$ are the predictions from the noise input $\mathbf{v}_i^x$, specified as follows

$$\mathbf{h}_i = \underbrace{f_L \circ f_{L-1} \circ \cdots \circ f_1}_{L \text{ times}}(\mathbf{v}_i^x) \quad (19)$$

$$\hat{\mathbf{v}}_i^y = \underbrace{g_1 \circ g_2 \circ \cdots \circ g_L}_{L \text{ times}}(\mathbf{h}_i) \quad (20)$$

$$\hat{\mathbf{v}}_i^z = \underbrace{\phi_1 \circ \phi_2 \circ \cdots \circ \phi_L}_{L \text{ times}}(\mathbf{h}_i) \quad (21)$$

where $\circ$ indicates function composition, $\mathbf{h}_i$ is the shared hidden representation from the triplet $\langle \mathbf{v}^x, \mathbf{v}^y, \mathbf{v}^z \rangle$, and the functions f_l, g_l, and ϕ_l are non-linear projections, such as sigmoid function. We ignore the underscript for parameters in mapping functions f_l, g_l, and ϕ_l, for $l = \{1, ..., L\}$ in the above equations. In our case, we use the same number of layers L to all branches for simplicity and clarity. Note that each branch in the deep structure network can have different number of layers and nodes, only if they keep the same dimetionality of the shared representation.

Multi-view multi-class object recognition: Assume that $\mathbf{v}^x$ is the input image contains different objects, and $\mathbf{v}^y$ is the vector to describe the views to catch the object with cameras, and $\mathbf{v}^z$ indicates which class the object belongs to. The purpose is to answer whether these additional views with the low level image features are helpful to improve the recognition accuracy. Thus, we propose the following objective

$$\begin{aligned}\theta &= \text{argmin}_\theta \mathcal{L}(\mathbf{v}^x, \mathbf{v}^y, \mathbf{v}^z; \theta) \\ &= \text{argmin}_\theta - \sum_{i=1}^{N} \mathbf{v}_i^z \log \hat{\mathbf{v}}_i^z + (1 - \mathbf{v}_i^z)\log(1 - \hat{\mathbf{v}}_i^z) \quad (22)\end{aligned}$$

where θ is the set of the weights in the 3-way deep architecture as before. And $\hat{\mathbf{v}}_i^z$ is the prediction from the image $\mathbf{v}_i^x$ and its view $\mathbf{v}_i^y$, specified as follows

$$\mathbf{h}^{(x2)} = \underbrace{f_{L-1} \circ \cdots \circ f_1}_{L-1 \text{ times}}(\mathbf{v}_i^x) \quad (23)$$

$$\mathbf{h}^{(y2)} = \underbrace{g_{L-1} \circ \cdots \circ g_1}_{L-1 \text{ times}}(\mathbf{v}_i^y) \quad (24)$$

$$\mathbf{h}_i = \text{sigmoid}(\mathbf{h}^{(x2)^T}\mathbf{W}^{(x3)} + \mathbf{h}^{(y2)^T}\mathbf{W}^{(y3)}) \quad (25)$$

$$\hat{\mathbf{v}}_i^z = \underbrace{\phi_1 \circ \phi_2 \circ \cdots \circ \phi_L}_{L \text{ times}}(\mathbf{h}_i) \quad (26)$$

where we ignore the bias term for $\mathbf{h}_i$ for clarity.

Given the parameter initialization with CD algorithm, we can minimize the objective function in Eq. 18 or 22 respectively to estimate the model parameters. To fine-tune the model, we compute the gradients w.r.t. weights via back-propagation in each layer in the objective function, and then we use any gradient-based methods to update the model parameters, such as L-BFGS [19].

Note that the joint visual restoration and labeling task is different from the multi-view multi-class recognition task. In fact, we can see the differences between the two objective functions in Eqs. 22 and 18, even though they used the same multimodal deep structure and initialized the model parameters with the same CD algorithm. The multi-view multi-class object recognition leverage multiple inputs to improve the classification performance, while the joint visual restoration and labeling learns a joint model for multiple outputs from only one input channel. The former has one input and two outputs, thus it is related to multi-task learning. While the latter has two inputs and one output prediction, by leveraging multi-source information for multi-classification problem.

3.4 Relationship to other models

We analyzed the differences between our model and other deep structures, such as deep autoencoder and deep Boltzmann machines.

3.4.1 deep autoencoder

The deep autoendcoder [1] can be thought as a bi-modal deep model with feed-forward networks. It consists of encoder and decoder in order to recover the data itself by learning the shared hidden representation. On the contrary, our model is a K-way deep feed-forward neural network with multiple channels and our model can adjust to different optimization problems given the same architecture, for example the joint visual restoration and labeling. Although these two models can use the same CD algorithm to initialize model parameters, our model is more powerful to handle multiple output predictions, instead of just recovering the data in the deep autoencoder.

3.4.2 deep Boltzmann machines

A multimodal DBM can be viewed as a composition of unimodal undirected pathways. Each path-way can be pretrained separately in a completely unsupervised fashion, which make it possible to leverage a large supply of unlabeled data. The middle graph in Fig. 1 is a 3-fan multimodal DBM. In our framework, we define a K-fan deep structure, where each branch can be a deep learning model, such as DBN, LSTM and CNN for different multimodal inputs. Thus, our model is more powerful. Moreover, the multimodal DBM is a generative model, while our model is a discriminative model. In other words, our modal is a multi-path feed-forward neural network with a shared representation, which can be optimized in an discriminative manner to handle different tasks.

3.4.3 Multi-task learning

Multi-task learning is an approach to learn a problem together with other related problems at the same time, using a shared representation. Our multiple deep neural network can predict multiple outputs by learning a shared representation for multi-task. Thus, our model can be used to handle multi-task learning problems, such as the joint visual restoration and labeling. In addition, our model can leverage multiple inputs or resources to improve the prediction or classification, such as the multi-view multi-class object recognition in our case. Thus, our deep K-fan structure is flexible and powerful, and can be optimized according to different tasks.

4 Experiments

As mentioned before, we test our deep model with two tasks: joint visual restoration and labeling, and multi-view multi-class object recognition. For the former task, we evaluated the performance with Peak signal-to-noise ratio (PSNR). In addition, we also used error rate to evaluate whether denoising is helpful or not in the recognition task. As for the multi-view multi-class object recognition, we leverage the lower level image features with additional multiple sources, such as camera views to improve the classification accuracy.

4.1 Data description

The MNIST dataset[1] consists of 28×28-size images of handwriting digits from 0 through 9 with a training set of 60,000 examples and a testing set of 10,000 examples, and has been widely used to test character denoising and recognition methods. A set of examples are shown in Fig. 2(a). In the experiment, we test both denoising and recognition performance. As for the noise model, we considered the structural noise that is hard to remove in the handwriting images. Basically, we random add two strokes to each digits, refer the structural noise in Fig. 2(b), which shows heavily corrupt images, with more than 50% regions.

The USPS Handwritten binary Alphadigits[2] are binary images with size 20×16 pixels. There are digits of "0" through "9" and capital "A" through "Z", with 39 examples of each class. In our experiments, we only test our method on the binary Alphabets.

The multi-view multi-class dataset[3] consists of 23 minutes and 57 seconds of synchronized frames taken at 25fps from 6 different calibrated DV cameras. One camera was placed about 2m high of the ground, two others where located on a first floor high, and the rest on a second floor to cover an area of $22m \times 22m$. This ground truth contains 242 annotated multi-view non-consecutive frames. These frames contain different real situations where pedestrians, cars and buses appear and can cause high occlusions among them. In our task, we want to test whether the additional multi-view information is helpful or not in the multi-classification task. Hence, we crop all objects in the frames according to its groundtruth bounding box, and then we resize them into the same size 88×64. Then, we vectorize all the cropped objects into 5632 dimensions, with additional 6 dimension camera view information to each object. Finally, we get 4907 instances, with 1295 persons, 3354 cars and 58 buses.

[1] http://yann.lecun.com/exdb/mnist/

[2] http://www.cs.nyu.edu/~roweis/data/binaryalphadigs.mat

[3] http://cvlab.epfl.ch/data/multiclass

4.2 Experimental setting

In all experiments, we use the 3-fan deep model with 3-layer nonlinear mapping in each pathway, with the learning rate 0.1 and CD-1 step sampling to initialize the model weights. We set the constant $\lambda = 1$ to weigh the two term in the objective function in Eq. 18. In the fine-tuning state, we used L-BFGS to update the model parameters. In particular, for the joint restoration and labeling, we used CD (multimodal DBMs) to initialize the weights, and then optimize Eq. 18 to fine-tune the model parameters with L-BFGS. For the MNIST digits and the USPS alphabets, we set the number of hidden nodes [400 200 250] for each pathway respectively in the 3-layer deep model.

As for the multi-view multi-class object recognition task, we used the same CD to initialize the weights by maximizing the joint likelihood, and then optimize Eq. 22 to fine-tune the model parameters. And we evaluate the performance with 10-fold cross validation. As for the model structure, we set the number of hidden nodes [400 200 250] in each pathway respectively in the 3-layer deep model.

4.3 Experimental results

We evaluated our method on both joint visual restoration and labeling and multi-view multi-class object recognition tasks.

4.3.1 Joint restoration and labeling

In this part, we tested our model on the multi-task learning problem: joint restoration and labeling by minimizing Eq. 18. Before analyzing the performance, we first evaluated the PSNR lower bound on the noise images, as well as the accuracy upper bound on the clear images.

PSNR and error rate bounds: (1) generating the noisy digits: we added noise to each MNIST image by randomly drawing two strokes to construct its noisy observation (clutter more than 50% regions). Thus, for all the clear training and testing digits, with 60,000 training and 10,000 testing images respectively, we can construct the corresponding noisy images. (2) training a classifier on the clear data: we first learned the deep neural network (DNN) [3] for classification tasks by minimizing the cross entropy. In the experiment on the MNIST digits, we used the default DDN parameters, namely 4-layer deep structure with hidden nodes [500 500 2000 10] respectively for each layer. Then we trained the model on the 60,000 clean images to learn the DNN classifier and then tested it on the 10,000 clean testing set. The error rate on the clean testing set we can get using DNN is 1.2%, while the error rates on the noisy testing set is 61.0% heavily corrupted noise in Fig. 2(b). And the PSNR lower bound on the noisy digits is 7.65 dB, which is calculated on the noisy testing set.

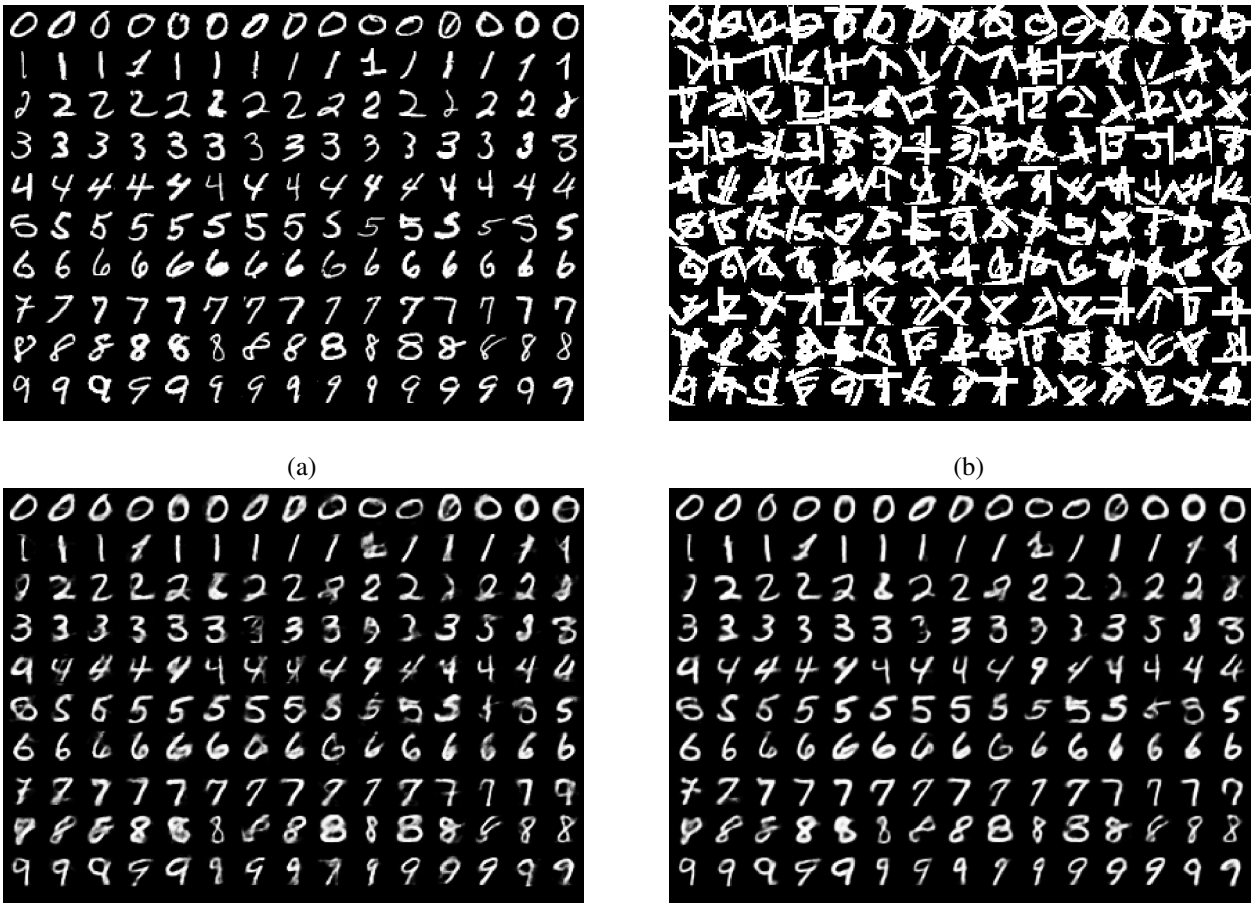
Model	PSNR (dB)	Error rate (%)
Wiener [20]	11.7	58.5
RoBM [14]	13.9	52.6
DAE	13.58	35.9
Our method	**18.6**	**12.7**
DNN [1]	≥ 7.65	$1.20 \sim 61.0$

Table 1: The experimental comparison on the heavily corrupted MNIST digits. It indicates that our method is better than competitive baselines on both denoising and recognition tasks.

Evaluation: To test our joint learning model, we trained our model on the 60,000 triplets: clean digit, noisy digit and its label. Then, we tested it on the 10,000 noisy testing dataset to restore its clear image and also predict its label. The denoised result (random sampled) of our model was shown in Fig. 2(d), and its quantitative result was shown in Table 1.

We compared our method to competitive baselines, which has separated pipelines, restoring image first and then recognizing it. Note that all the quantitative results of baselines are evaluated on the denoised images using the DNN classifier. More specifically, for each baseline, we first use it to denoise images, and then we use the DNN classifier to evaluate the recognition accuracy on the denoised images. The denoising result with denoising autoencoder (DAE)[12] is shown in Fig. 2(c), and its recognition accuracy is shown in Table 1. And it clearly demonstrates that our joint learning method is better than DAE and ROBM [14].

We also evaluated our method on the USPS alphabets. Similar to the experiment on the MNIST digits, we added random strokes to the alphabets to create the noisy observations. Because there are only 39 training images for each class, we generated 10 corrupted samples for each clean image. In the experiments, we divided the total 39×26 binary images into training set (accounting for 80%) and testing set (the rest 20%), and we learned a 2-layer DNN model with 100 and 64 hidden nodes in each layer respectively on the clean training set. The classification performance on the clean testing set is 1.29% in Table 2, while the error rate on the corresponding noisy testing set is 67.4%. Then we used the learned DNN model to evaluate the denoising performance for the baselines. The visual performance of denoise autoencoder is shown in Fig. 3(c), and the result of our model is shown in Fig. 3(d). The quantitative comparison between our method and the baselines is shown in Table 2, which demonstrates our method for joint visual restoration and labeling is better than competitive baselines on both denoising and recognition tasks.



(a) (b) (c) (d)

Figure 2: The denoising results comparison on the heavily occluded MNIST digits. (a) original images; (b) noisy images with random structures; (c) denoising results with denoising autoencoder; (d) denoising results with our joint restoration and labeling model.

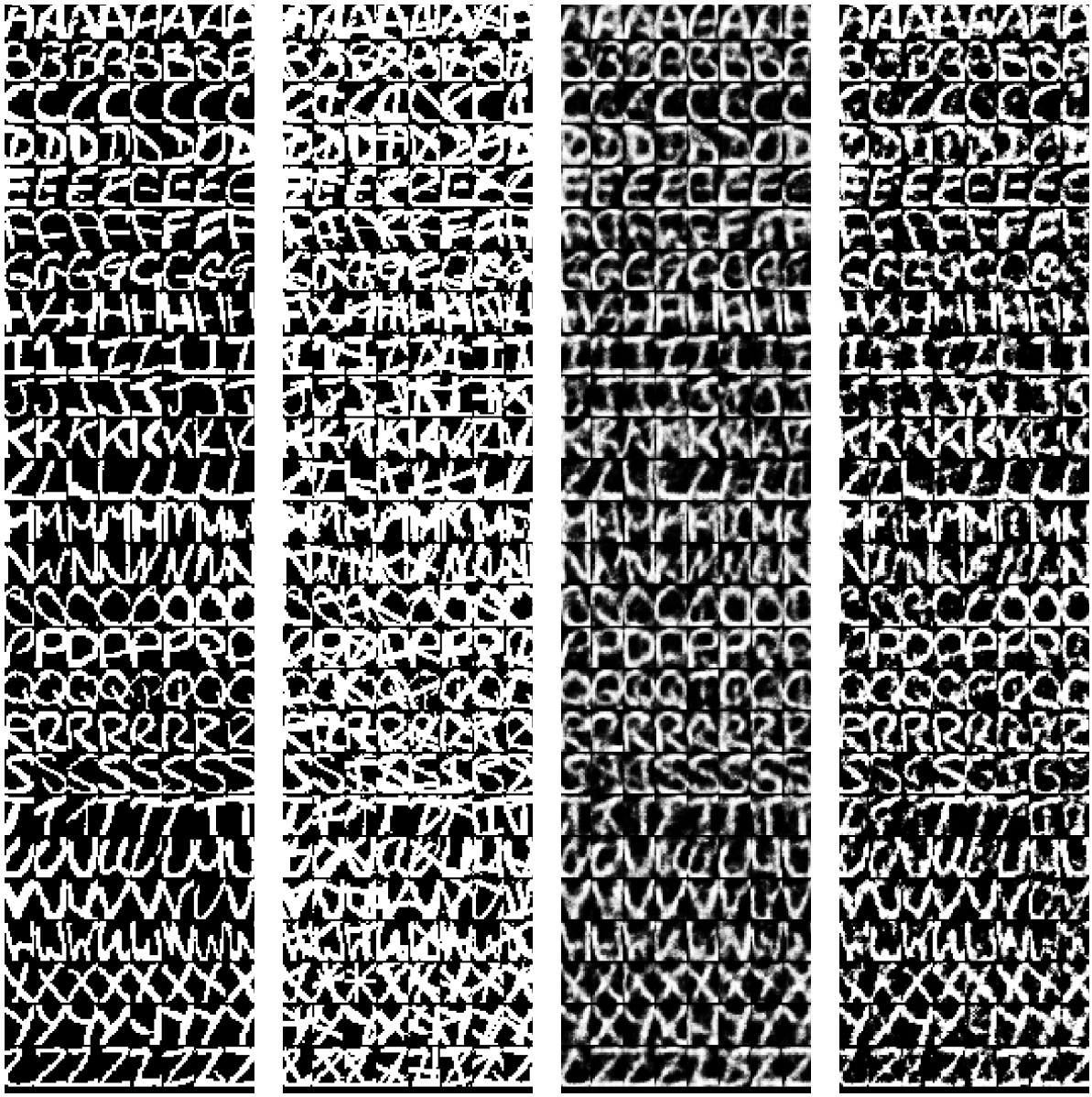
Model	PSNR (dB)	Error rate (%)
Wiener [20]	14.2	67.8
RoBM [14]	16.3	62.8
DAE	19.2	42.5
Our method	**19.6**	**32.8**
DNN [1]	≥ 8.12	$1.29 \sim 67.4$

Table 2: The experimental comparison on USPS alphabets. The PSNR value of DNN is 8.12 dB, which shows the lower bound on the noisy testing set. The error rate using DNN means that the error rates on the noisy testing set and the original clean testing set are 67.4% and 1.29% respectively. It demonstrates that our method is better than competitive baselines on both denoising and recognition tasks.

4.3.2 Multi-view multi-class recognition

In this part, we test our model on the multi-view multi-class recognition task. In this task, our model has two inputs and one output prediction, thus it can leverage multiple sources to boost classification accuracy.

The multi-view multi-class dataset contains total 4907×5632 instances, belonging to 3 classes. And there are additional 6 bits camera view information for each instance. This multi-class dataset is unbalanced with 1295 persons, 3354 cars and 58 buses. The purpose is to answer whether the additional camera information is helpful or not in the multi-classification tasks. In our experiment, we use 10-fold cross validation, by randomly sampling 9 fold for training and the rest for testing. To train our 3-fan multimodal deep model, we can leverage both lower level image features and multi-view information as two inputs, and predict the object class in the output. We optimize our model via Eq. 22 and show our result in Table 3. The SVM with multi-view yields better result than SVM w/o multi-view, which indicates that the additional multi-view information is helpful. And our model outperforms multimodal DBM and SVM, which clearly demonstrates that our 3-fan deep model can effective leverage multiple sources to boost per-



(a) (b) (c) (d)

Figure 3: The denoising results comparison on USPS alphabet (the type 2 noise). (a) original images from 'A' to 'Z' arranged in the top-down manner; (b) noisy images with random structures; (c) denoising results with denoising autoencoder; (d) denoising results with our 3-fan deep model.

formance. It also indicates our discriminative model is better than multimodal DBM in the classification problem.

To sum up, we can optimize different objective functions to achieve different goals. For example, if we want to handle multi-task learning, we can use the objective function in Eq. 18 for multiple outputs. If we want to leverage multiple sources to boost classification performance, we can propose a similar objective function as in Eq. 22. However, no matter what objective function we use, we still use the same K-fan deep multimodal structure. Thus, our model is flexible and powerful to leverage multiple inputs for multiple predictions.

Model	Error rate (%)
SVM w/o multi-view	9.59
SVM w multi-view	6.92
Multimodal DBM [5]	7.10
Our model	**4.80**

Table 3: The experimental comparison on the multi-view multi-class recognition dataset. It demonstrates that our model outperforms other methods significantly.

5 Conclusions

In this paper, we propose a generalized K-fan deep structure, with shared representations for multimodal learning. Our deep multimodal structure is powerful because each branch can be a deep learning model for different inputs. Given the deep model, we can optimize different objective functions to learn the shared representation from multimodalities to handle different tasks. To learn the model parameters, we take a two stage steps: parameter initialization and parameter fine-tuning. In the parameter initialization stage, we use the CD algorithm to maximize the joint likelihood as the multimodal DBM does. In the fine-tuning stage, we update the model parameter according to the defined objective function. We test our model on two tasks: the joint restoration and labeling, and the multi-view multiclass object recognition. The former task is to handle the multi-task learning problem, while the latter is to answer whether our model can leverage multiple sources to boost classification performance. The experimental results show our K-fan deep structure model is flexible and powerful, and can be optimized according to different functions to address different problems.

References

[1] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[2] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning." in *ICML*, 2011, pp. 689–696.

[3] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[4] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *TPAMI*, 2012.

[5] N. Srivastava and R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," *Journal of Machine Learning Research*, vol. 15, pp. 2949–2980, 2014.

[6] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 1980.

[7] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[9] E. P. Xing, R. Yan, and A. G. Hauptmann, "Mining associated text and images with dual-wing harmoniums," in *In Conference on Uncertainty in Artificial Intelligence*, 2005.

[10] M. J. Huiskes, B. Thomee, and M. S. Lew, "New trends and ideas in visual concept detection: The mir flickr retrieval evaluation initiative," in *Proceedings of the International Conference on Multimedia Information Retrieval*, ser. MIR '10. New York, NY, USA: ACM, 2010, pp. 527–536. [Online]. Available: http://doi.acm.org/10.1145/1743384.1743475

[11] M. Guillaumin, J. J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification." in *CVPR*. IEEE, 2010, pp. 902–909.

[12] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *JMLR*, 2010.

[13] L. Holmström and P. Koistinen, "Using additive noise in back-propagation training," Rolf Nevanlinna Institute, Research Reports A3, 1990.

[14] Y. Tang, R. Salakhutdinov, and G. Hinton, "Robust boltzmann machines for recognition and denoising," in *CVPR*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 2264–2271.

[15] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 854–869, May 2007. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2007.1055

[16] R. Salakhutdinov and G. E. Hinton, "An efficient learning procedure for deep boltzmann machines," *Neural Computation*, vol. 24, no. 8, pp. 1967–2006, 2012.

[17] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy." in *NIPS*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds. Morgan Kaufmann, 1993, pp. 3–10.

[18] R. M. Neal and G. E. Hinton, "Learning in graphical models," M. I. Jordan, Ed. Cambridge, MA, USA: MIT Press, 1999, ch. A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants, pp. 355–368.

[19] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM J. Sci. Comput.*, vol. 16, no. 5, pp. 1190–1208, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1137/0916069

[20] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964.